adding to each transition leaving "p" a non-empty-string transition, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the automaton A.

~~32~~ 31. (NEW) The computer readable medium of claim ~~31~~ 30, the method on the computer readable medium further comprising:

removing inaccessible states using a depth-first search of the automaton A.

~~33~~ 32. (NEW) The computer readable medium of claim ~~31~~ 30, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

~~34~~ 33. (NEW) The computer readable medium of claim ~~31~~ 30, wherein the step of computing ε-closure for each input state of an input automaton A further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

~~35~~ 34. (NEW) A circuit programmed to operate a method of removing an empty string term from an automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the automaton A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition, wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to a state "q" in the automaton A.

~~36~~ 35. (NEW) The circuit of claim ~~35~~ 34, the method programmed into the circuit further comprising:

removing inaccessible states using a depth-first search of the automaton A.

~~37~~ 36. (NEW) The circuit of claim ~~35~~ 34, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

~~38~~ 37. (NEW) The circuit of claim ~~35~~ 34, wherein the step of computing ε-closure for each input state of an input automaton A further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

~~39~~ 38. (NEW) A computer readable medium programmed to operate a method of removing an empty string term from a transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the transducer A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the transducer A.

39 ~~40~~. (NEW) The computer readable medium of claim ~~39~~ 38, the method on the computer readable medium further comprising:

removing inaccessible states using a depth-first search of the transducer A.

40 ~~41~~. (NEW) The computer readable medium of claim ~~39~~ 38, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

41 ~~42~~. (NEW) The computer readable medium of claim ~~39~~ 38, wherein the step of computing ε-closure for each input state of an input transducer A further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

42 ~~43~~. (NEW) A circuit programmed to operate a method of removing an empty string term from a transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the transducer A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the transducer A.

43 ~~44~~. (NEW) The circuit of claim 42 ~~43~~, the method programmed into the circuit further comprising:

removing inaccessible states using a depth-first search of the transducer A.

44 ~~45~~. (NEW) The circuit of claim 42 ~~43~~, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

45 ~~46~~. (NEW) The circuit of claim 42 ~~43~~, wherein the step of computing ε-closure for each input state of an input transducer A further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

46 ~~47~~. (NEW) An automaton B having no ε-transitions, the automaton B produced according to a method of removing the ε-transitions from an input automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the input automaton A.

47 ~~48~~. (NEW) The automaton of claim 46 ~~47~~, the method of creating the automaton B further comprising:

removing inaccessible states using a depth-first search of the input automaton.

48 ~~49~~. (NEW) The automaton of claim 46 ~~47~~, wherein the step of adding to E[p] non-ε-transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

49 ~~50~~. (NEW) A automaton of claim 46 ~~47~~, wherein the step of computing an ε-closure for each input state of an input automaton A further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

50 ~~51~~. (NEW) A transducer B having no ε-transitions, the transducer B produced according to a method of removing the ε-transitions from an input transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input transducer;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the input transducer A.

51 ~~52~~. (NEW) The automaton of claim 50 ~~51~~, the method of creating the transducer B further comprising:

removing inaccessible states using a depth-first search of the input transducer.

52 ~~53~~. (NEW) The automaton of claim 50 ~~51~~, wherein the step of adding to E[p] non-ε-transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

53 ~~54~~. (NEW) A automaton of claim 50 ~~51~~, wherein the step of computing an ε-closure for each input state of an input transducer A further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

54 ~~55~~. (NEW) A computer readable medium storing an executable automaton B having no ε-transitions, the automaton B produced according to a method of removing ε-transitions from an input automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to a state "q" in the input automaton.

55 ~~56~~. (NEW) A computer readable medium storing an executable transducer B having no ε-transitions, the transducer B produced according to a method of removing ε-transitions from an input transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and